POLITECNICO
MILANO 1863

Wireless Internet Project

Crowd counter through sniffing Wi-Fi packets

Andrés Felipe Rodríguez Vanegas – 915761
Gianpaolo Alesiani - 915405

A.Y. 2018/2019

1. Introduction

The big popularity of mobile devices and all different types of other wireless gadgets as smartphones, laptops and tablets, has required the introduction of one big protocol for all wireless technologies, IEEE 802.11 or Wi-Fi.
The fast developing of Wi-Fi has led to important advantages, especially in comfort and movement with these devices. However, due to his peculiar open nature, Wi-Fi exchanges same messages between access point and client completely in clear, introducing on the other side very dangerous drawbacks. Luckily, most of the time communication messages are encrypted, so only who has the proper key can read the packets.
We want to put our attention in these in-clear messages; we can extract from them many information such as the MAC address of the transmitter, the vendor of the devices, Wi-Fi fingerprints that can reveal different type of information about the owner.
We will discover, with the help of a Wi-Fi adapter, the total list of devices that can be listened by our antenna, their Received Signal Strength Indicator (RSSI), how many devices are present between the Wi-Fi adapter and a chosen smartphone or laptop (which can be seen as how many wireless devices we find inside in a room).
Final step of our research consists in the analysis of the vendors devices nearby, building a graph that could represent their percentual.

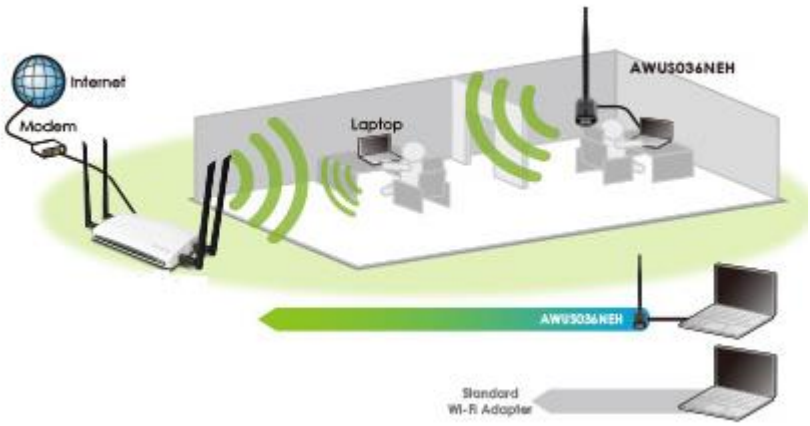
2. Background

Sniffing is the technique that let us capture and analyze the data transmitted inside a Wi-Fi network, a useful way to check all messages exchanged, so that to increase the security of the network.
However, sniffing can also be used with malicious intent in order to catch personal information such as password and banking data without any permissions. In the 802.11 protocol two different types of message are sent in clear:
Beacons: transmitted by the access point (AP), they are the management frames that contain information about the network, useful to announce the presence of a wireless LAN and to synchronize the members of the service set. Beacons are sent with a fixed period at a time called Target Beacon Transmission Time (TBTT), and a beacon interval of duration 102.4 milliseconds [2].
Probe requests: special frames sent by a client station requesting information from either a specific access point specified by SSID (in this case is called “directed” probe request), or all access points in the area (the broadcast probe request). With these messages mobile devices can request information from access points and accelerate the Wi-Fi connection process. Since they are broadcasted before association with APs, they are sent in the clear.

Wi-Fi probe requests are unique identifiers, as they contain the MAC address of mobile devices, and may also include a list of preferred Wi-Fi networks accessed by these devices in the past.

3. Monitor mode and choice of the antenna

Since probe requests are sent in clear, they are easy to collect with commodity hardware (such as wireless cards in monitoring mode) and standard software, like Airodump, TCPDump, or Wireshark. Monitor mode allows a hardware to monitor all traffic received on a wireless channel. Unlike managed mode (in this configuration the card only cares about the traffic that affects his system), monitor mode allows packets to be captured without having to associate with an access point or ad hoc networks first.
During the testing of the project, three different wireless network adapters have been used, since the first two did not satisfy our requests. Firstly, we tried to capture packets with the wireless card of the computer; however, wireless card in laptops usually do not support monitor mode, therefore, it was not possible using the chipset of our Wi-Fi card in our laptop, to set no other than a basic Wi-Fi connection. An external network adapter was crucial in order to capture as many packets as possible.
The choice falls into TP-LINK TL-WN722N, that was highly recommended in blogs of hacking and Wi-Fi security; unfortunately, the version of our antenna (the third), and the chipset of this adapter did not support monitor mode, only the first version did. Consequently, we changed our adapter: Alfa AWUS036NEH, a long range, stable, fast, and well supported b/g/n wireless network adapter, that works in the frequency of 2.4GHz.



This time the device was suitable for what we were looking for: monitor mode is well supported, and the range satisfies the purpose of this project.

4. Description of the project

Software used for this work is Wireshark. With Wireshark we can translate the raw “0” and “1” that our device listens, in human readable information and organize them in a graphic interface.
First part of the Python code consists in sniff and capture the probe requests packets using the Wi-Fi adapter (defined as *wlan1 in Linux*) and the report of this research is written in a

.cap file, that we call *probe-req-a|b|c.cap*. In the code it is added a filter, that let us receive only probe request packets.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	b2:d1:eb:b9:44:67	Broadcast	802.11	144	Probe Request, SN=3765, FN=0, Flags=..
2	0.020012869	Arcadyan_a2:74:ba	Broadcast	802.11	95	Probe Request, SN=2167, FN=0, Flags=..
3	0.039081930	Arcadyan_a2:74:ba	Broadcast	802.11	95	Probe Request, SN=2168, FN=0, Flags=..
4	0.122708742	Apple_74:b7:18	Broadcast	802.11	121	Probe Request, SN=231, FN=0, Flags=...
5	0.146011658	Apple_74:b7:18	Broadcast	802.11	121	Probe Request, SN=232, FN=0, Flags=...
6	0.187191637	Apple_74:b7:18	Broadcast	802.11	121	Probe Request, SN=235, FN=0, Flags=...
7	0.467188883	IntelCor_97:45:9b	Broadcast	802.11	88	Probe Request, SN=855, FN=0, Flags=...
8	0.543767064	Apple_50:82:58	Broadcast	802.11	147	Probe Request, SN=1141, FN=0, Flags=..
9	0.570256632	IntelCor_97:45:9b	Broadcast	802.11	88	Probe Request, SN=859, FN=0, Flags=...
10	1.055785742	da:4b:41:66:ce:e1	Broadcast	802.11	167	Probe Request, SN=348, FN=0, Flags=...

Afterwards it is defined the time of the capture. About this choice, it is significative pointing out that a time too short would cause the escape of some packets we could be interested into. However, we noticed during tests that *tshark* stops when it receives too many packets. Despite this problem, its use is preferred to *airodump-ng*, since this last software has a different management of the packets. Useful information can be found in [1]: in average one probe requests every minute is broadcasted by the Android (operative systems we have used for our tests) and a lower number for iOS devices.
Subsequently we analyze the probe requests in capture file; source address, RSSI and the SSID are extracted and broadcast probe requests are added in the arrays.
Meantime, it has been defined the MAC address of a chosen wireless device, defined as “known_mac” (our known_mac will be always 60:14:66:43:0c:8e, smartphone we used as tester); we will use this information to set the radius of the circle in which we want to count the number of wireless devices. This implementation will have an immediate response also from our sight: in this way is easy to check if capture time is too short (less devices are detected than the effective presents) or approximately correct.
Next part consists in the discovery of the devices captured by the antenna: in output we get the different MAC address, sorted in hexadecimal order, the corresponding RSSI in average (since different packets can be intercepted for the same MAC address) and the number of devices within the range of the antenna. We terminate this part of the code with the output of the signal power of the “known_mac” and the devices inside its range.
Following step consists in finding the vendors (we used part of the code used in class of the analyze_probe_requests program): we exploit the fact that the first 3 bytes of each MAC address are assigned to the manufacturer, extract the information of the vendors, from the oui.txt file, and then find the vendor of each MAC found in the range.
Final part of the code consists in plotting histograms of RSSI for all the devices and plotting the pie of vendors.

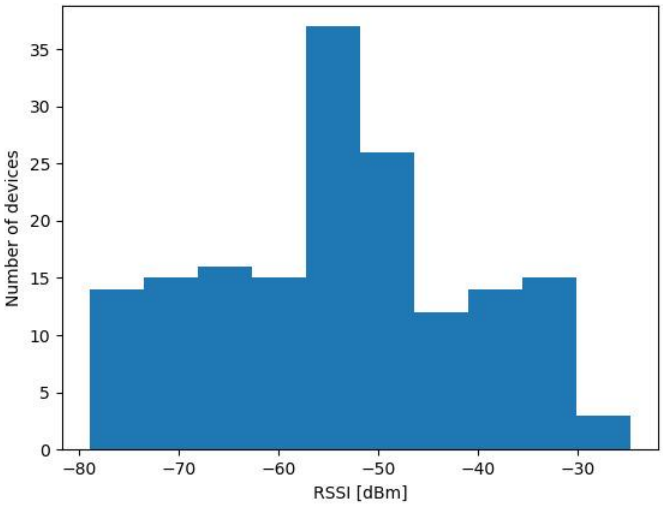
5. Results of the tests

In this chapter we are going to describe the three tests that we have performed describing the three different configurations:

a) With the first test we detect the total number of devices that are listened by our antenna and we want to prove that leaving our smartphone strictly close to the transmitter, number of devices identified in the range is zero.

Results are represented in the following pictures.

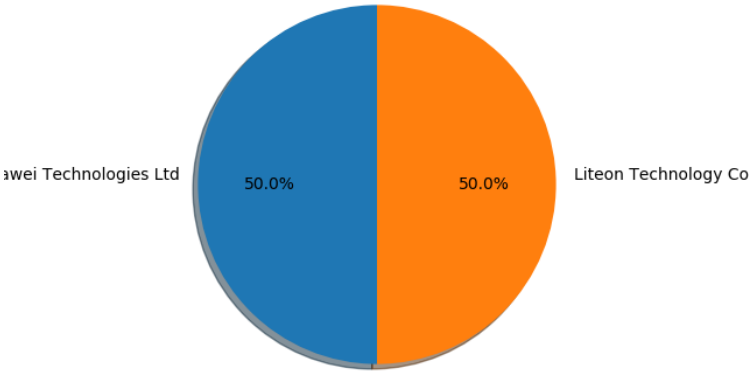
```
----------------------
Limit RSSI: -24.6

MAC               RSSI

There were 0 devices within the area of the selected MAC address (60:14:66:43:0c:8e)
```



We can read the limit power, -24.6 dBm; it coincides with the power of the "known_MAC". No devices with greater power are found inside this range, as expected.
With the histogram we report the number of devices presents in each range of the power. Thanks to this antenna, more than 170 different MAC address have been detected.

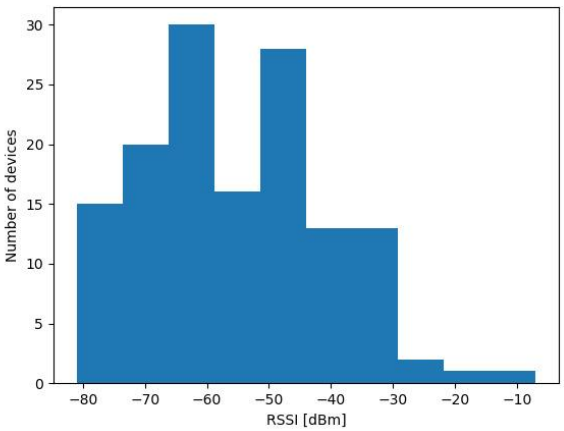
b) In the second trial we make the radius of the circle larger by leaving the "known_MAC" just a meter away from the antenna. Direct consequence will be a lower limit power.
In this case results can be demonstrated easily; just two devices must be found inside the range: the closest MAC address is the one of the network card of the laptop where antenna is attached and this will be always present in the captures. The other is one of our smartphones that we have left inside the range. In the next picture these results are reported.

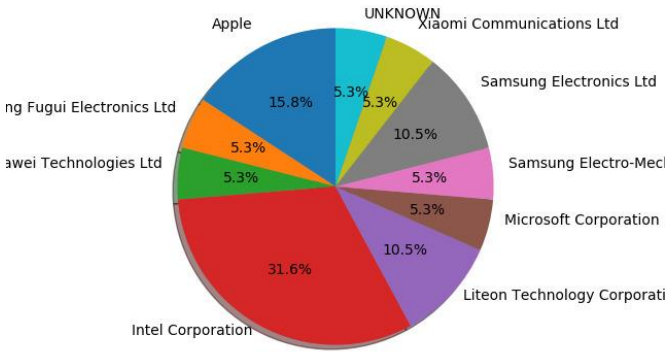
```
----------------------
Limit RSSI: -25.181818181818183

MAC               RSSI
00:5a:13:b7:b8:4a -23.0
50:5b:c2:b6:67:99 -25.0

There were 2 devices within the area of the selected MAC address (60:14:66:43:0c:8e)
```



Second picture reports the pie chart of the result.

c) Third and last trial is the investigation of the number of wireless devices present inside the room and the representation of their vendors in a graph.
This time, “known_MAC” is put in the corner of the room we are in, while we have conducted the experiment in the center of the classroom. In this way we find in ideal condition to count smartphones and laptops around us.
Results are quite satisfactory since, in a not crowd classroom, we detected almost 30 devices, a number that is appropriate with the environment.
As conclusion, two graphs are reported.





The first one represents number of devices in distances, such as the one present in the first test. The second one is a pie chart with the representation of the vendors.

6. Conclusion

With this project we wanted to see more in detail a part of the program of Wireless Internet we were particularly interested into and touch with hands different topics covered in lessons and deal with problems of real life. Results were quite in line with our expectations: first trial was just to check the correct operation of the code we wrote.
The second trial let us check if, in an environment where we already knew the devices inside the range, the answers of the python code were as expected.
With the third and last trial we performed the research of the number of people inside a classroom.

7. Bibliography

[1] Short: How Talkative is your Mobile Device? An Experimental Study of Wi-Fi Probe Requests

[2] https://mrncciew.com/2014/10/08/802-11-mgmt-beacon-frame/

[3] A. Redondi et al. “Passive Classification of WiFi enabled devices” – MSWIM 2016